

REMARKS

In a final Office Action dated December 5, 2006, the Examiner rejected claims 1, 2 and 5-8 under 35 U.S.C. §103(a) as unpatentable over Baumgartner et al. (US 6,108,764) in view of Arimilli et al. (US 2003/0009643); and rejected claims 3-4 and 9-22 under 35 U.S.C. §103(a) as unpatentable over *Baumgartner* and *Arimilli* in view of Carpenter et al. (US 6,115,804).

Applicants respectfully traverse the Examiner's rejections.

In response to the previous office action, applicants explained certain issues with respect to the background of and motivation for their invention. This discussion remains relevant to the present rejections, and is incorporated by reference in this response. In addition to repeating some of the previous response, applicants wish to emphasize certain additional points regarding the design of a conventional cache coherency directory and the reason for the use of set associativity.

In a NUMA architecture, a respective discrete subset of main memory is usually associated with each node, processor or group of processors. Nodes or processors may additionally cache arbitrary portions of main memory, and in particular may cache memory from other nodes. Various mechanisms exist to support cache coherency in such an environment. In particular, a node having a local discrete subset of main memory typically may maintain a cache line state directory with respect cache lines from the local subset which are stored in caches, and particularly in caches of other nodes and devices.

The purpose of a cache line state directory is to determine whether data of interest in the local portion of main memory is in some cache and/or certain cache state information with respect to the cached data. Memory data is accessed using real memory addresses. Therefore, the cache line state directory receives a real memory address as input, and is designed to provide an output indicating whether the data at the input real memory address is in cache, and possibly certain additional state information. Making this determination is often a performance critical path.

Docket No.: ROC920030390US1
Serial No.: 10/760,431

It is therefore desirable to provide a mechanism for rapidly determining cache state from a given real memory address input. Conventionally, this is achieved using a well-known set-associative directory, in which a portion of the real memory address is hashed to directly access an associativity set containing several entries, and the entries of the associativity set are examined in parallel by logic associated with the cache line state directory to determine which, if any, of the entries corresponds to the input real memory address.

A set-associative directory provides rapid access when the entries of the associativity set can be examined in parallel by hardware logic. Separate hardware comparators and other logic are required for each entry in the associativity set, so as the associativity set becomes larger, the amount of hardware support required increases proportionately. This fact places practical constraints on the size of the associativity set.

It would, of course, be possible to have arbitrarily large associativity sets, in which the entries are examined sequentially, but this would substantially increase the time required to determine the cache line state for a given real address, time which lies in a performance critical path. The same is true of a simple table directory which is not indexed. It must either be examined sequentially (requiring long access time), or it must be examined with parallel hardware (requiring a large amount of hardware).

For all of these reasons, conventional cache line state directories are generally constructed as set-associative directories, in which the associativity set is accessed by hashing the real memory address.

As explained in response to the previous office action, applicants observed a problem with a conventional set-associative cache line state directory where certain special-purpose caches (such as a cache for an I/O bridge device) have a very high turnover in a relatively small cache. As a result of the high turnover, a large number of cache load operations occurring in a short time tends to displace references in the cache line state directory to cache lines stored in processor

caches. This causes various inefficiencies in the use of cache line state directory space, in the need to send invalidation messages to the I/O bridge device, and so forth.

Since the cache in an I/O bridge device or certain similarly behaving devices is often quite small, applicants address this problem by providing a separate cache line state directory portion, in which the entries do not correspond to real addresses (as in the case of a set associative cache), but instead have a one-to-one correspondence to the slots in the device cache. Thus, the number of references required to be kept for the device cache can never exceed the number of slots in the device cache (unlike the case of the conventional set-associative cache). Such a design is practical for a small, high-turnover device, because it is possible to construct hardware logic to examine each cache line state directory entry in the portion allocated to the device, since it has relatively few entries. Such a design would be far less attractive for a relatively large processor cache.

In response to the previous office action, applicants amended claims to clarify the nature of the one-to-one correspondence between entries in the cache line state directory and cache lines of the device cache. Applicants further pointed out that while *Baumgartner* and *Carpenter* disclose some form of cache line state directory in a general sense, they do not discuss in detail the structure of such directories, and that their directories are presumably organized along conventional lines, i.e., as set-associative directories. There is, at least, no disclosure in *Baumgartner* or *Carpenter* of a cache line state directory or portion thereof in which the entries have a one-to-one correspondence to cache lines.

All independent claims herein recite that a portion of the cache line state directory has a fixed one-to-one mapping of entries with slots in the device cache. Certain claims (e.g., independent claim 14 and dependent claims 3, 12 and 21) recite additionally that a separate portion of the cache line state directory contains entries for the processor cache or caches, and some of these claims recite additionally that entries in the processor portion correspond to real

addresses of data (i.e., different indexing is used in different portions of the cache line state directory). None of these features are taught or suggested by the cited art.

Applicants' representative claim 1 recites:

1. A digital data processing system, comprising:
 - a memory;
 - at least one processor having at least one associated cache for temporarily caching data from said memory;
 - at least one device having a device cache, said device cache having a fixed number of slots for caching data, said fixed number being greater than one, each slot caching a cache line of data; and
 - a cache coherency mechanism, said cache coherency mechanism including a cache line state directory structure, *said cache coherency mechanism selectively determining whether to send cache line invalidation messages to said at least one device using state information in said cache line state directory structure*, wherein at least a portion of said cache line state directory structure associated with said at least one device contains exactly said fixed number of cache line entries, *each entry having a fixed correspondence to a unique respective one of said fixed number of slots* for caching data of said device cache. [emphasis added]

In the latest action, the Examiner cites *Arimilli* in addition to the previously cited art. *Arimilli* discloses a NUMA system architecture having multiple processors in each node, each processor having its own portion of memory and one or more caches. *Arimilli*'s invention appears to be directed to the reduction of queue length in a node controller, by delaying placing certain memory access operations on the queue. Specifically, a memory access generated by a processor within a node is not placed on the queue for remote operations until all local processors have responded and the requested data has not been found in the memory portions or caches associated with the local processors. *Arimilli* further discloses a cache coherence mechanism, including a respective "local memory directory" associated with each processor. The local memory directory is a form of cache line state directory, i.e., it is used to maintain cache line state information for data in the memory portion associated with the processor, and which might be held in a cache of

another processor in the same or a different node. *Arimilli*'s local memory directory appears to be set-associative; at least, there is no disclosure of one-to-one correspondence between the entries of the local memory directory and entries in the caches of other processors or devices.

In the rejection, the Examiner cites a passage from *Arimilli* which discloses a cache directory structure. The cache directory is generally conventional in design, being set-associative and accessed by hashing an address. There is a one-to-one correspondence between entries of the cache directory and cache lines of the cache, as in conventional design. Indeed, it could hardly be otherwise, for the purpose of the directory is to identify what is in the lines of the cache.

The Examiner reasons that one-to-one correspondence of a directory and a cache structure is thus shown, and therefore it would have been obvious to combine this design feature with the cache coherency structures disclosed in *Baumgartner* and *Carpenter*. Applicants respectfully disagree.

The Examiner apparently concedes that *Arimilli*'s cache directory is not a "cache line state directory" as recited in applicants' claims, i.e., it lacks recited functional features of a "cache line state directory", e.g., is not used to determine whether to send invalidation messages to the device. The Examiner appears to be relying on the teaching of a property of one-to-one correspondence in a structure which maintains information relative to a cache.

Applicants believe that this rejection misses the inventive aspect of their invention. Applicants concede that a logic designer of ordinary skill in the art could have designed a cache line state directory having a one-to-one correspondence of selective entries with cache lines of a device cache, ***given the proper motivation and direction to do so***. The point is that such a motivation is not disclosed in the prior art. It is disclosed in only one place: applicants' specification. The Examiner is therefore using hindsight from applicants' specification to construct the claimed invention from known elements or features.

As explained above, the motivation for applicants' invention is to avoid a situation in which the cache line state directory becomes filled up with entries relating to an I/O bridge or similar device, crowding out other entries. The entries for the I/O bridge are typically very short lived, so that at any given instant the vast majority of such entries would be already invalidated. If a cache line state directory is thus filled with I/O bridge entries pointing to invalidated lines, it becomes largely useless as a performance enhancement mechanism.

Applicants recognize this problem and design around it by separating the entries for cache lines stored in the high-turnover I/O bridge device cache from entries for cache lines in the relatively more stable processor caches. With respect to the former, there is a one-to-one correspondence between entries in the cache line state directory and cache lines in the device. Some additional hardware is needed to examine the entries for the device in parallel, but this is a manageable amount because the number of such entries is small.

This is not an issue of whether it is proper to combine the references, but a question of what the combination of references suggests. There is nothing whatsoever in any of the cited art which teaches or suggests the nature of the problem described above. Nor is there anything which teaches or suggests the solution employed by applicants. On the contrary, all three references appear to employ conventional set-associative structures for maintaining such cache coherency information. To the extent the references teach anything with respect to cache coherency mechanisms, they teach away from applicants' invention.

For all the reasons stated above, the claims are patentable over *Baumgartner*, *Arimilli* and/or *Carpenter*.

In view of the foregoing, applicants submit that the claims are now in condition for allowance, and respectfully request reconsideration and allowance of all claims. In addition, the Examiner is encouraged to contact applicants' attorney by telephone if there are outstanding issues left to be resolved to place this case in condition for allowance.

Respectfully submitted,

DUANE A AVERILL, et al.



By: _____

Roy W. Truelson

Registration No. 34,265

Telephone: (507) 202-8725

Docket No.: ROC920030390US1
Serial No.: 10/760,431